

## 509334 v3

1           5.       The method of claim 4, further comprising:  
2       determining said minimum-hop path to said destination node by:  
3           traversing a one of said rows corresponding to said destination node  
4               from a first column of said columns to a second column of said  
5               columns, and  
6       storing path information representing said minimum-hop path while  
7           traversing said data structure from said second column to said  
8           first column, said second column being a first one of said  
9           columns encountered when traversing said one of said rows  
10          from said first column to said second column having non-  
11          default cost entry.

1           7.       The method of claim 4, further comprising:  
2       determining said minimum-cost path to said destination node by:  
3           identifying a minimum-cost column of said columns, said minimum-  
4           cost column having a lowest cost entry of all of said columns in  
5           a one of said rows corresponding to said destination node, and  
6       storing path information representing said minimum-cost path while  
7           traversing said data structure from said minimum-cost column  
8           to a first column of said columns.

509334 v3

1 9. A method of finding a path in a network comprising:  
2 creating a path table, wherein:  
3 said path table comprises a first number of rows and a second number  
4 of columns,  
5 said network comprises a plurality of nodes and a plurality of links,  
6 each one of said plurality of nodes is coupled to at least one other of  
7 said plurality of nodes by at least one of said plurality of links,  
8 and  
9 said path begins at a root node of said plurality of nodes;  
10 processing each row in a first column of said second number of columns, said  
11 processing each row in said first column of said second number of  
12 columns resulting in said first column containing corresponding first  
13 connectivity information; and  
14 processing each remaining column, wherein  
15 said each remaining column is a one of said second number of columns  
16 other than said first column, and  
17 said processing each remaining column resulting in said first column  
18 containing corresponding subsequent connectivity information  
19

1 10. The method of claim 9, wherein said processing said each row in said  
2 first column of said second number of columns comprises:  
3 for said each row in said first column of said second number of columns,  
4 wherein a selected node of said plurality of nodes corresponds to said  
5 row in said first column,  
6 if said selected node is a neighbor of said root node,  
7 storing a first link cost in a first cost entry, wherein  
8 said first link cost is a link cost of a first one of said plurality of  
9 links,  
10 said first one of said plurality of links is between said root node and  
11 said selected node, and

12 said first cost entry is a cost entry of said row in said first column,  
13 storing a root node identifier in a first previous node entry,  
14 wherein said root node identifier represents said root  
15 node and said first previous node entry is a previous  
16 node entry of said row in said first column, and  
17 storing a node identifier representing said selected node in a  
18 storage area,  
19 else,  
20 storing a maximum cost value in said first cost entry, and  
21 storing a null value in said first previous node entry.

1 11. The method of claim 10, wherein said processing said each remaining  
2 column comprises:  
3 for said each remaining column,  
4 if said storage area is not empty,  
5 copying each row of a preceding column to a corresponding  
6 row of said remaining column, said preceding column  
7 being a one of said second number of columns other  
8 than said remaining column,  
9 for each stored node identifier, said stored node identifier being  
10 stored in said storage area and corresponding to a  
11 current node of said plurality of nodes,  
12 removing said stored node identifier from said storage area,  
13 for each neighboring node, said neighboring node being a neighbor  
14 of said current node,  
15 adding a neighboring link cost to a preceding path cost in order  
16 to yield an alternate path cost, wherein  
17 said neighboring link cost is a link cost of a second one of  
18 said plurality of links, said second one of said  
19 plurality of links being between said current node  
20 and said neighboring node, and

21 said preceding path cost is a cost value stored in a cost entry  
22 of a row of said preceding column, said row of said  
23 preceding column corresponding to said current  
24 node,

25 if said alternate path cost is less than a cost value stored in a  
26 current cost entry,

27 storing said alternate path cost in said current cost entry,  
28 said current cost entry being a cost entry of a row of  
29 said remaining column, said row of said remaining  
30 column corresponding to said neighboring node,  
31 storing a node identifier representing said current node in a  
32 previous node entry of said row of said remaining  
33 column, and  
34 storing a node identifier representing said neighboring node  
35 in said storage area.

1 12. The method of claim 11, further comprising:  
2 identifying said root node, wherein said root node stores a topology database  
3 comprising:  
4 connectivity information regarding which ones of said plurality of  
5 nodes are coupled to which other ones of said plurality of  
6 nodes, and  
7 a link cost for each one of said plurality of links.

1 13. The method of claim 12, further comprising:  
2 generating said topology database by:  
3 for each one of said plurality of nodes,  
4 identifying, at least one neighboring node that is a neighbor of  
5 said each one of said plurality of nodes and at least one  
6 neighboring link that couples said at least one  
7 neighboring node to said each one of said plurality of  
8 nodes, wherein said plurality of nodes includes said at

9                   least one neighboring node and said plurality of links  
10                  includes said at least one neighboring link,  
11                  determining a link cost for each one of said at least one  
12                  neighboring link, and  
13                  storing said link cost at said each one of said plurality of nodes;  
14                  and  
15         distributing said connectivity information for each one of said plurality of  
16         nodes to other of said plurality of nodes.

1           14.     The method of claim 13, wherein said link cost is a physical length of a  
2     corresponding one of said plurality of links.

1           15.     The method of claim 13, wherein said link cost is a bandwidth of a  
2     corresponding one of said plurality of links.

1            16.     The method of claim 11, wherein said first number is equal to a  
2     number of nodes in said plurality of nodes.

1            17.    The method of claim 11, wherein each one of said second number of  
2    columns corresponds to a number of hops.

1           18.     The method of claim 17, wherein said number of hops is equal to a  
2     maximum number of hops.

1            19.     The method of claim 18, wherein said maximum number of hops is a  
2     maximum number of hops possible when using said method.

1           20.     The method of claim 18, wherein said maximum number of hops is a  
2     maximum number of hops selected by a user.

1            21.     The method of claim 11, wherein said each neighboring node is one of  
2     said plurality of nodes that is a neighbor of said current node.

1           22.     The method of claim 11, wherein said stored node identifier is already  
2 stored in said storage area at a time when processing of said remaining column is  
3 begun.

1           23.     The method of claim 11, wherein said preceding path cost is a cost of a  
2 path between said root node and said current node.

1           24.     The method of claim 11, wherein said storage area is a queue.

1           25.     The method of claim 11, further comprising:  
2 selecting a destination node from said plurality of nodes, said destination node  
3 being a one of said plurality of nodes other than said root node; and  
4 determining a minimum number of hops between said root node and said  
5 destination node by  
6 setting said minimum number of hops to one, and  
7 for each one of said second number of columns, incrementing said  
8 minimum number of hops by one if a cost entry in a row of said  
9 one of said second number of columns is equal to said  
10 maximum cost value, said row of said one of said second  
11 number of columns corresponding to said destination node.

1           26.     The method of claim 11, further comprising:  
2 selecting a destination node from said plurality of nodes, said destination node  
3 being a one of said plurality of nodes other than said root node; and  
4 determining a minimum cost of said path, said path being between said root  
5 node and said destination node, by searching a one of said first number  
6 of rows corresponding to said destination node for a one of said second  
7 number of columns having a smallest cost entry of cost entries of said  
8 second number of columns in said one of said first number of rows.

1           27.     The method of claim 11, wherein each one of said first number of rows  
2 corresponds to a one of said plurality of nodes.

- 1
- 2
- 3
- 4

- 1
- 2
- 3

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8



said intermediate column is a one of said second number of  
columns between said first column and said first non-  
maximum-cost column, inclusive, and  
said first non-maximum-cost column is a first one of said  
second number of columns, when proceeding from said  
first column to said first non-maximum-cost column, for  
which a cost entry in a row corresponding to said  
destination node is not said maximum cost value,  
storing, in said path storage area, a previous node identifier stored in a  
previous node entry of a row of said intermediate column,  
wherein said row of said intermediate column corresponds to a  
one of said plurality of nodes represented by said current node  
identifier, and  
setting a current node identifier to said previous node identifier.

32. The method of claim 28, further comprising:  
selecting a destination node from said plurality of nodes, said destination node  
being a one of said plurality of nodes other than said root node;  
storing a destination node identifier in said path storage area, said destination  
node identifier representing said destination node;  
setting a current node identifier to said destination node identifier;  
searching a one of said first number of rows corresponding to said destination  
node for a minimum path-cost column, wherein  
said minimum path-cost column is a one of said second number of  
columns having a smallest cost entry of said second number of  
columns in said one of said first number of rows; and  
for each intermediate column, proceeding from said minimum path-cost  
column to said first column, wherein said intermediate column is a one  
of said second number of columns between said first column and said  
minimum path-cost column, inclusive,  
storing, in said path storage area, a previous node identifier stored in a  
previous node entry of a row of said intermediate column,

18 wherein said row of said intermediate column corresponds to a  
19 one of said plurality of nodes represented by said current node  
20 identifier, and  
21 setting a current node identifier to said previous node identifier.

1 33. The method of claim 11, wherein said link cost is a physical length of a  
2 corresponding one of said each one of said at least one neighboring links.

1 34. The method of claim 11, wherein said link cost is a bandwidth a  
2 corresponding one of said each one of said at least one neighboring links is configured  
3 to carry.

1 35. The method of claim 11, wherein:  
2 said storing said node identifier representing said current node in said previous  
3 node entry of said row of said remaining column further comprises:  
4 storing a node identifier stored in a next node entry of said row of said  
5 preceding column in a next node entry of said row of said  
6 remaining column; and  
7 said method further comprises:  
8 for each row in said first column,  
9 if said corresponding node is a neighbor of said root node,  
10 storing a node identifier representing said corresponding node in a  
11 first next node entry, wherein said first next node entry is a  
12 next node entry of said row in said first column, and  
13 else,  
14 storing a null value in said first next node entry.

1 36. A method of finding a path in a network comprising:  
2 creating a path vector, wherein:  
3 said path vector comprises a first number of rows,  
4 said network comprises a plurality of nodes and a plurality of links,

5 each one of said plurality of nodes is coupled to at least one other of  
6 said plurality of nodes by at least one of said plurality of links,  
7 and  
8 said path begins at a root node of said plurality of nodes;  
9 for a first hop of a maximum number of hops, processing each row in said first  
10 number of rows for said first hop, a selected node of said plurality of  
11 nodes corresponding to said row;  
12 for each remaining hop of said maximum number of hops, processing said  
13 each row in said first number of rows for said each remaining hop;

1 37. The method of claim 36, wherein said processing each row in said first  
2 number of rows for said first hop comprises:  
3 for said first hop of said maximum number of hops,  
4 for each row in said first number of rows, a selected node of said  
5 plurality of nodes corresponding to said row,  
6 if said selected node is a neighbor of said root node,  
7 storing a first link cost in a first cost entry, wherein  
8 said first link cost is a link cost of a first one of said plurality of  
9 links,  
10 said first one of said plurality of links is between said root node  
11 and said selected node, and  
12 said first cost entry is a cost entry of said row, and  
13 storing a node identifier representing said selected node in a  
14 storage area,  
15 else  
16 storing a maximum cost value in said first cost entry.

509334 v3  
38. The method of claim 37, wherein said processing said each row in said  
2 first number of rows for said each remaining hop comprises:  
3 for said each remaining hop of said maximum number of hops,  
4 if said storage area is not empty,

5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30

1  
2

1  
2

1           41.     The method of claim 38, wherein said maximum number of hops is a  
2     maximum number of hops selected by a user.

1           42.     The method of claim 38, wherein said each neighboring node is a one  
2     of said plurality of nodes that is a neighbor of said current node.

1           43.     The method of claim 38, wherein said stored node identifier is already  
2     stored in said path storage area at a time when processing of said remaining column is  
3     begun.

1           44.     The method of claim 38, wherein said preceding path cost is a cost of a  
2     path between said root node and said current node.

1           45.     The method of claim 38, wherein said storage area is a queue.

1           46.     A computer system comprising:  
2     a processor coupled to a network, wherein said network comprises a plurality  
3         of nodes and a plurality of links and each one of said plurality of nodes  
4         is coupled to at least one other of said plurality of nodes by at least one  
5         of said plurality of links;  
6     computer readable medium coupled to said processor; and  
7     computer code, encoded in said computer readable medium, configured to  
8         cause said processor to find a path in said network by virtue of being  
9         configured to cause said processor to:  
10        generate at least one path cost data set, said path cost data set  
11           representing a path cost between a root node of said nodes and  
12           destination node of said nodes, wherein said path begins at said  
13           root node and ends at said destination node; and  
14        access said at least one path cost data set wherein  
15           said generating and said accessing are performed in such a  
16           manner that a minimum-hop path and a minimum-cost

17 path can be determined from said at least one path cost  
18 data set,

19 said minimum-hop path represents a path between said root  
20 node and said destination node having a minimum  
21 number of hops, and

22 said minimum-cost path represents a path between said root  
23 node and said destination node having a minimum cost.

1 47. The computer system of claim 46, wherein said computer code is  
2 further configured to cause said processor to:  
3 store said at least one path cost data set in a path storage area such that said at  
4 least one path cost data set can be accessed to determine said  
5 minimum-hop path and said minimum-cost path.

1 48. The computer system of claim 47, wherein said computer code is  
2 further configured to cause said processor to:  
3 allocate said path storage area in a data structure that facilitates said access to  
4 determine said minimum-hop path and said minimum-cost path.

1 49. The computer system of claim 46, wherein said computer code is  
2 further configured to cause said processor to:  
3 store said at least one path cost data set in a data structure, wherein  
4 said data structure is a two-dimensional array of entries arranged in a  
5 plurality of rows and a plurality of columns,  
6 each one of said rows in said data structure corresponds to one of said  
7 plurality of nodes, and  
8 each one of said columns in said data structure corresponds to a given  
9 hop count.

1           50.    The computer system of claim 49, wherein said computer code is  
2 further configured to cause said processor to:  
3           determine said minimum-hop path to said destination node by virtue of being  
4           configured to cause said processor:  
5           traverse a one of said rows corresponding to said destination node from  
6           a first column of said columns to a second column of said  
7           columns, and  
8           store path information representing said minimum-hop path while  
9           traversing said data structure from said second column to said  
10          first column, said second column being a first one of said  
11          columns encountered when traversing said one of said rows  
12          from said first column to said second column having non-  
13          default cost entry.

1           51.    The computer system of claim 50, wherein said first column  
2 corresponds to said root node.

1           52.    The computer system of claim 49, wherein said computer code is  
2 further configured to cause said processor to:  
3           determine said minimum-cost path to said destination node by virtue of being  
4           configured to cause said processor:  
5           identify a minimum-cost column of said columns, said minimum-cost  
6           column having a lowest cost entry of all of said columns in a  
7           one of said rows corresponding to said destination node, and  
8           store path information representing said minimum-cost path while  
9           traversing said data structure from said minimum-cost column  
10          to a first column of said columns.

1           53.    The computer system of claim 52, wherein said first column  
2 corresponds to said root node.

1           54.    A computer program product for finding a path in said network,  
2 wherein said network comprises a plurality of nodes and a plurality of links and each  
3 one of said plurality of nodes is coupled to at least one other of said plurality of nodes  
4 by at least one of said plurality of links and said computer program product is encoded  
5 in computer readable media and comprising:

6           a first set of instructions, executable on a computer system, configured to  
7           generate at least one path cost data set, said path cost data set  
8           representing a path cost between a root node of said nodes and  
9           destination node of said nodes, wherein said path begins at said root  
10          node and ends at said destination node; and

11          a second set of instructions, executable on said computer system, configured to  
12          access said at least one path cost data set wherein  
13          said generation and said access are performed in such a manner that a  
14          minimum-hop path and a minimum-cost path can be  
15          determined from said at least one path cost data set,  
16          said minimum-hop path represents a path between said root node and  
17          said destination node having a minimum number of hops, and  
18          said minimum-cost path represents a path between said root node and  
19          said destination node having a minimum cost.

1           55.    The computer program product of claim 54, further comprising:  
2           a third set of instructions, executable on said computer system, configured to  
3           store said at least one path cost data set in a path storage area such that  
4           said at least one path cost data set can be accessed to determine said  
5           minimum-hop path and said minimum-cost path.

1           56.    The computer program product of claim 55, further comprising:  
2           a fourth set of instructions, executable on said computer system, configured to  
3           allocate said path storage area in a data structure that facilitates said  
4           access to determine said minimum-hop path and said minimum-cost  
5           path.



1 57. The computer program product of claim 54, further comprising:  
2 a third set of instructions, executable on said computer system, configured to  
3 store said at least one path cost data set in a data structure, wherein  
4 said data structure is a two-dimensional array of entries arranged in a  
5 plurality of rows and a plurality of columns,  
6 each one of said rows in said data structure corresponds to one of said  
7 plurality of nodes, and  
8 each one of said columns in said data structure corresponds to a given  
9 hop count.

1 58. The computer program product of claim 57, further comprising:  
2 a fourth set of instructions, executable on said computer system, configured to  
3 determine said minimum-hop path to said destination node, said fourth  
4 set of instructions comprising:  
5 a first subset of instructions, executable on said computer system,  
6 configured to traverse a one of said rows corresponding to said  
7 destination node from a first column of said columns to a  
8 second column of said columns, and  
9 a second subset of instructions, executable on said computer system,  
10 configured to store path information representing said  
11 minimum-hop path while traversing said data structure from  
12 said second column to said first column, said second column  
13 being a first one of said columns encountered when traversing  
14 said one of said rows from said first column to said second  
15 column having non-default cost entry.

1 59. The computer program product of claim 58, wherein said first column  
2 corresponds to said root node.

1        60.    The computer program product of claim 57, further comprising:  
2        a fourth set of instructions, executable on said computer system, configured to  
3        a first subset of instructions, executable on said computer system,  
4        configured to determine said minimum-cost path to said destination  
5        node, said fourth set of instructions comprising:  
6        identify a minimum-cost column of said columns, said minimum-cost  
7        column having a lowest cost entry of all of said columns in a  
8        one of said rows corresponding to said destination node, and  
9        a second subset of instructions, executable on said computer system,  
10        configured to store path information representing said  
11        minimum-cost path while traversing said data structure from  
12        said minimum-cost column to a first column of said columns.

1        61.    The computer program product of claim 60, wherein said first column  
2        corresponds to said root node.

*add A'* }